ESD-TR-75-371

MTR-3102

COMMUNICATIONS PROCESSORS: CATEGORIES, APPLICATIONS, AND TRENDS

MARCH 1976

Prepared for

DEPUTY FOR COMMAND AND MANAGEMENT SYSTEMS
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
Hanscom Air Force Base, Bedford, Massachusetts

Approved for public release; distribution unlimited.

Project No. 572E

Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts

Contract No. F19628-75-C-0001

REVIEW AND APPROVAL

This technical report has been reviewed and is approved for publication.

STANLEY P. DERESKA, Colonel, USAF
Chief, Techniques Engineering
Division

DONALD P. ERIKSEN
Project Manager
Techniques Engineering Division

FOR THE COMMANDER

FRANK J. EMMA, Colonel, USAF
Director, Information Systems
Technology Applications Office
Deputy for Command & Management Systems

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER ESD-TR-75-371	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) COMMUNICATIONS PROCESSORS: CATEGORIES, APPLICATIONS, AND TRENDS	5. TYPE OF REPORT & PERIOD COVERED
	6. PERFORMING ORG. REPORT NUMBER MTR-3102
7. AUTHOR(s) K. T. Mahoney	8. CONTRACT OR GRANT NUMBER(s) F19628-75-C-0001
9. PERFORMING ORGANIZATION NAME AND ADDRESS The MITRE Corporation Box 208 Bedford, MA 01730	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Project No. 572E
11. CONTROLLING OFFICE NAME AND ADDRESS Deputy for Command and Management Systems Electronic Systems Division, AFSC Hanscom Air Force Base, Bedford, MA 01731	12. REPORT DATE MARCH 1976 13. NUMBER OF PAGES 54
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

COMMUNICATIONS
COMPUTER NETWORKS
COMPUTERS
FRONT—END PROCESSORS

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Communications processor applications are expanding as the trend continues towards distributed processing in information systems. This report identifies and describes the major categories of communications processors. As an example of current applications, the use of communications processors within existing systems

is detailed. The potential impact of emerging technologies on the design and use of communications processors is discussed. Tradeoffs and guidelines in selecting a device for a specific application are included, with manufacturer models presented as illustrations.

ACKNOWLEDGMENTS

The author wishes to thank James P. Hanks for his assistance and support during the production of this document.

This report has been prepared by The MITRE Corporation under Project No. 572E. The contract is sponsored by the Electronic Systems Division, Air Force Systems Command, Hanscom Air Force Base, Massachusetts.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

SECTION I

INTRODUCTION

With the increase of distributed processing in Air Force information systems, the applications of communications processors have been rapidly expanding. Many network control functions will be off-loaded from the larger host computers. In some cases, emergency service may be provided by the communications processor to handle traffic when the host computer is out of service. As the demands on existing systems increase, communications processors may be added to augment the system's capabilities. To determine when and where a communications processor might be used, a knowledge of the capabilities and potential applications of the available designs is essential.

Accordingly, the purpose of this report is to provide the reader with information on where communications processors can be applied, show examples of actual applications, and give some guidelines in choosing a communications processor for a specific application. Towards this end the report treats four major topics. Section II details the communications processor categories, including the functions and components of each class of processor. Section III presents examples of the applications of communications processors within existing systems. Section IV looks at the potential impact of protocol standardization, microprocessors, and higher order language standardization on communications processors and their use within computer networks. Section V deals with some guidelines in selecting a specific communications processor for an application, with type examples given as illustrations.

SECTION II

COMMUNICATIONS PROCESSOR CATEGORIES

There are three major communications processor categories by function: front-end processing, message-switching, and remote concentration. This section presents the features, functions, and capabilities of each category.

FRONT-END PROCESSING

Front-end processing is the most widespread use of programmable communications processors. The primary purpose of a front-end processor (FEP) is for communications control; that is, the FEP handles all the communications between the host computer and the individual terminals. In this paper, "terminal" denotes a teletype, display, remote batch entry station, or even a remote computer. Early FEP devices were hard-wired controllers, configured for specific line types and incapable of network processing or easy modification. The programmable FEP (PFEP) replaces and extends the hard-wired controller, interfacing the data network to the host computer.

Features

Flexibility and economy have been the basis for the increased popularity of the PFEP. Inherent in its programmable nature is the user's ability to modify the software. As system requirements change, (for example, line speeds and data transmission codes), the PFEP can be modified to handle the new configuration. A PFEP can introduce into the system an added measure of reliability not possible with a hard-wired controller because:

- The host computer is no longer burdened with network overhead functions.

- The PFEP provides a fail-soft feature. In the event of host failure, the PFEP can maintain reduced operation, providing limited processing and storage for incoming traffic on its own or a shared storage device. When the host returns on-line, backlogged traffic can be forwarded for processing.

Security considerations, such as required code words or station designations, can be handled by both hard-wired and programmable FEPs; however, the software of the PFEP allows it to handle more easily any complex checking.

For the purposes of this discussion, the term FEP will refer to programmable processor except where specifically noted.

Functions

An FEP controls the incoming data lines and handles formatting and message control functions. In a typical configuration, routine and non-productive overhead can be off-loaded from the host processor, reducing the time spent by the host on servicing I/O interrupts. Fully code-converted and verified message blocks can be assembled and queued in the FEP memory and transferred to the host like the data from any high-speed peripheral device. During busy periods, an FEP can dump incoming data to a mass storage device and complete the handling when the load decreases. For example, in the delta configuration (1) (Figure 1), both the FEP and host have access to the same mass storage files; if the host computer should malfunction, the FEP can continue to receive message traffic from the network and store it for later host processing.

Several functions are necessary in any data communications system, whether or not front-end processing is used. As an aid in evaluating the usefulness of an FEP in a particular system, the major data communications functions which an FEP may take over from the host computer are detailed:

- Line Control - The FEP handles I/O between the host and the attached terminals. In addition to the control functions for establishing connections and switching transmission directions, there may be data synchronizing functions. Connections may be point-to-point, switched, or multidrop. Line control procedures differ for these connection techniques, and may involve polling for control over the transmitting devices.

- Error Control - The FEP can detect and correct errors before they enter the host. A parity or cyclic redundancy check scheme is often used on the data.

- Message Assembly - Since the FEP provides a single high-speed line to the host, it assembles message blocks from the characters received from the terminals. Single characters and short message blocks may be packed into single larger messages for transmission to the host, thereby decreasing the time the host must dedicate to I/O handling.

- Message Buffering - When the host is unable to handle additional input, the FEP can hold messages until host processing time becomes available.

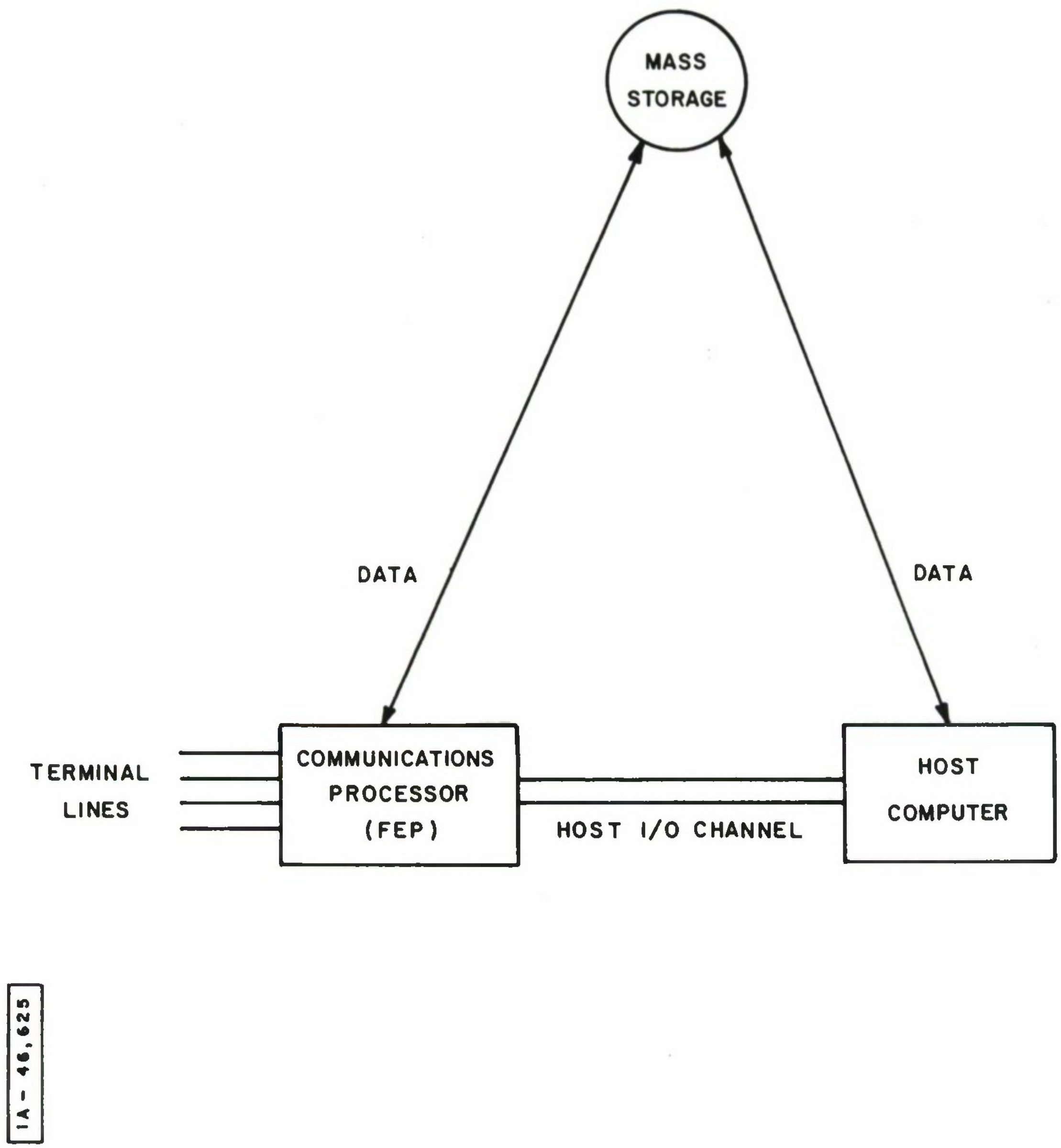


Figure 1 DELTA CONFIGURATION

- Code Conversion - The ability of the FEP to convert from one data code to another is an important factor in judging system flexibility. The FEP must be capable of handling differently coded lines while supplying messages with a single code to the host.

- Line Monitoring - An FEP can keep a record of the efficiency of the communication line operation, maintaining message traffic and error statistics.

A front-end processor can be configured to handle many varied functions in a particular network. The extent to which each function is implemented in a particular FEP depends on the individual device's size and capabilities, the scope of the software provided, and the particular application within the system.

Components

An FEP consists of the following major hardware components: processor and memory, host interface, communications multiplexor, and line interface units. Figure 2 shows the interconnection of the components within the FEP. Each component is described below:

- Processor and Memory - The processor handles interrupts from the incoming data lines and assembles the data for transfer to the host. Memory size generally ranges from 4K to 256K bytes.

- Host Interface - This interface allows the FEP to communicate directly on a standard I/O channel, thus appearing as another high-speed peripheral device.

- Communications Multiplexor - This device interfaces the line interface units to the FEP.

- Line Interface Units - These units interface the individual lines to the FEP multiplexor. Each unit is selected specifically for the line speed and transmission characteristics of a particular data line.

In addition to the hardware components, the FEP requires suitable software, dependent on the particular application and line handling requirements.

FEP Example

The following example illustrates the concept and advantages of front-end processing. A particular system utilizes no FEP, so each terminal interfaces directly to the host computer. Each time data is to be input by one of the terminals, the host must interrupt its

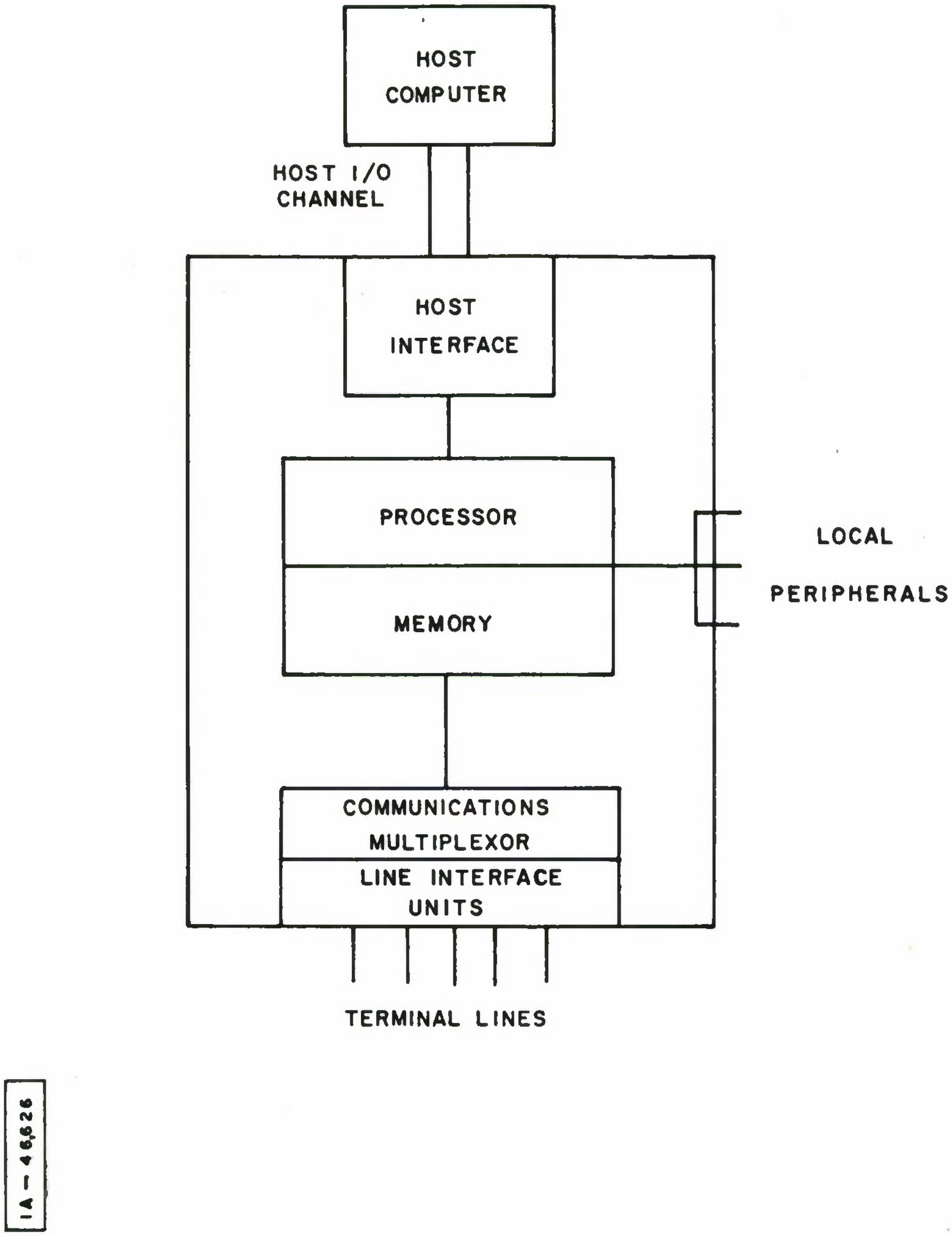


Figure 2 FEP COMPONENTS

processing to allow for I/O handling. As the number of terminal interrupts increases, so does the amount of time the host CPU must dedicate to the I/O interrupt handling, causing a corresponding reduction in the data processing time available. A large mainframe computer when not using an FEP typically dedicates from 15 to 30 percent of its time to communications handling.(2) On the other hand, consider the system where a front-end processor is used to handle the incoming data lines. The FEP can be programmed to receive data from a number of terminals and format it into a single message to be sent to the host computer. Since the host computer must now only be concerned with the data incoming from the FEP, the time required for I/O interrupt handling will be significantly reduced and the amount of data which the host can process will increase. Typically, by utilizing an FEP, the host communications overhead can be reduced to about 1 to 4 percent of its processing time.

Problem Areas

Despite the many advantages of an FEP, some problem areas must be considered. Since the FEP handles the data communications function and relieves the host computer of this burden, the possibility exists that as more data lines are added into the system, the FEP is given more responsibility for data handling than it can manage. Configuration guidelines for an FEP design must not be exceeded. With only a limited memory and computing capability, overloading the FEP could result in a data loss during periods of peak activity. Auxiliary storage devices can be configured in an FEP design to store peak load traffic, with catch-up processing during off-peak times.

Software is another potential problem area. As the FEP software becomes more complex, and thus larger, the storage available for processing incoming data is reduced. The capabilities of any software supplied for an FEP must be carefully analyzed to insure suitability for handling the expected network traffic and the ability to interface easily with the host computer as well as with the terminals. Large amounts of time spent to correct deficiencies in the FEP software can often negate any advantage which might be gained from a programmable FEP over a hard-wired device. Close inspection of the FEP software is necessary to insure it is capable of meeting the expected requirements. A savings in initial cost outlay may be more than erased by time-consuming software interfacing problems. This can be particularly true where the FEP is supplied by a manufacturer other than that of the larger host computer.

MESSAGE-SWITCHING

A communications processor in a message-switching application receives messages from remote terminals, or other message switches, analyzes them to determine their proper destination, performs any necessary code conversion, and transmits them to other remote terminals or message switches. As previously noted, a terminal within this context can include a teletype, display, remote batch entry station, or remote computer. The early uses of programmable processors in data communications networks were primarily in message-switching applications where they served as message handlers and dispatchers. Unlike front-end processors, which normally transmit data into a large scale computer for information processing, the message switch is generally a central clearinghouse for messages between all points in the communications network, and it does not process or react to the data content of the message.

Method of Operation

Message-switching systems use store-and-forward techniques, in which the message switch stores a complete message before forwarding it to the proper destination. Store-and-forward systems are message-oriented rather than character-oriented; that is, a store-and-forward system is concerned with complete messages from single remote stations, whereas other applications, such as remote concentration, may transmit message blocks consisting of only parts of the messages from several remote stations. In this system, a message originator can transmit data at its own speed to the message switch, without waiting for a direct connection to the ultimate destination. This method differs from circuit switching, where a circuit is established between sender and receiver for direct message transmission. If the originator has several messages ready for transmission, these can be sent to the message switch one after the other without reestablishing the connection, resulting in maximum communications line utilization. It is then the message switch's responsibility to route the messages to their destinations along available channels. The message switch can handle multiple-address messages efficiently, since the transmission to each addressee is independent and is under the message switch's control. The originator need only forward the multiple-address message once, designating all addressees. Sending and receiving stations need not have terminal equipment of the same line speeds and data formats, as the message switch can convert the messages from one code to another as necessary. Although the store-and-forward message-switching concept is not complex, it usually requires the largest communications processors and large memory buffers to handle the many alternatives of routing, queueing, priority assignment, and other control actions.

Message Handling

A switched message usually contains a header, consisting of the addressees and additional control characters as required (such as for priority), the text, and an end-of-message block. Incoming messages are placed into buffer storage to be collected into a standard block length. When the input lines are handled cyclically, a number of different messages may be undergoing assembly at any one time. After receiving the header, the message switch determines what type of handling should occur on this message. This handling might include the conversion of the data into a universal code used throughout the system, or the placing of the message into a priority queue by precedence code. In addition, the destination addressees must be identified, since, in larger systems, some of the destinations may require additional routing through other message-switching centers. Once the entire message has been received and stored, including its priority, time of arrival, and addressing information, the input line from the terminal is released for further use. When an output line to the addressee or next forwarding center becomes available, the transmission begins in a manner similar to the above. Having begun the output transmission by sending the beginning of the message from the output buffer, the subsequent message blocks are transferred at the receiving station's data rate. This allows a continuous transfer of the complete message. When transmission is required between message-switching centers, it is done over high-speed lines in order to minimize transmission delays. High traffic volume between centers would make higher speed lines cost-effective.

Since it is often possible to have multi-addressed messages, the output lines of the message switch will be more occupied than its input lines during peak periods. Hence, messages may require queuing to optimize the throughput of the available output channels. The queue generally consists of descriptive information regarding the destination, priority, time of arrival, and other details of a particular message. The queue indicates the next message to be transmitted. One method often used is maintaining a master queue of pointers to the actual message texts. This type of queue requires that messages be stored only once, whereas a queue using the actual message texts may require the storage of a single message more than once, when multiple addressing is used. When high priority messages are present, they must be able to bypass the queued messages and obtain priority transmission. Therefore, the queue must be capable of reorganizing as different priority messages are received. The master queue arrangement handles reorganization more efficiently, since only the pointers, not the complete messages, would be subject to position changes. Any overflow occurring in peak periods must be queued to auxiliary storage to prevent message loss. If auxiliary storage space is unavailable, input lines must be controlled to prevent any overflow.

Of major importance in a message-switching network is the ability of the communications processor to insure the correct transmission of a message to its destination, including retransmission if a message has not been correctly received. This reliability can normally be accomplished by requiring an acknowledgement from a receiving station for each message sent, and, in the event of a fault in the message storage area, a secondary storage unit to allow retrieval of the lost information. Secondary storage also provides a journal to maintain a copy of all message traffic for record purposes. Journals are usually maintained for a specified length of time.

Components

A message switch generally consists of the following major hardware components: processor and main message store, auxiliary storage devices, local peripheral devices, and communications line adapters. Figure 3 shows the interconnection of these components. The functions of each component follow:

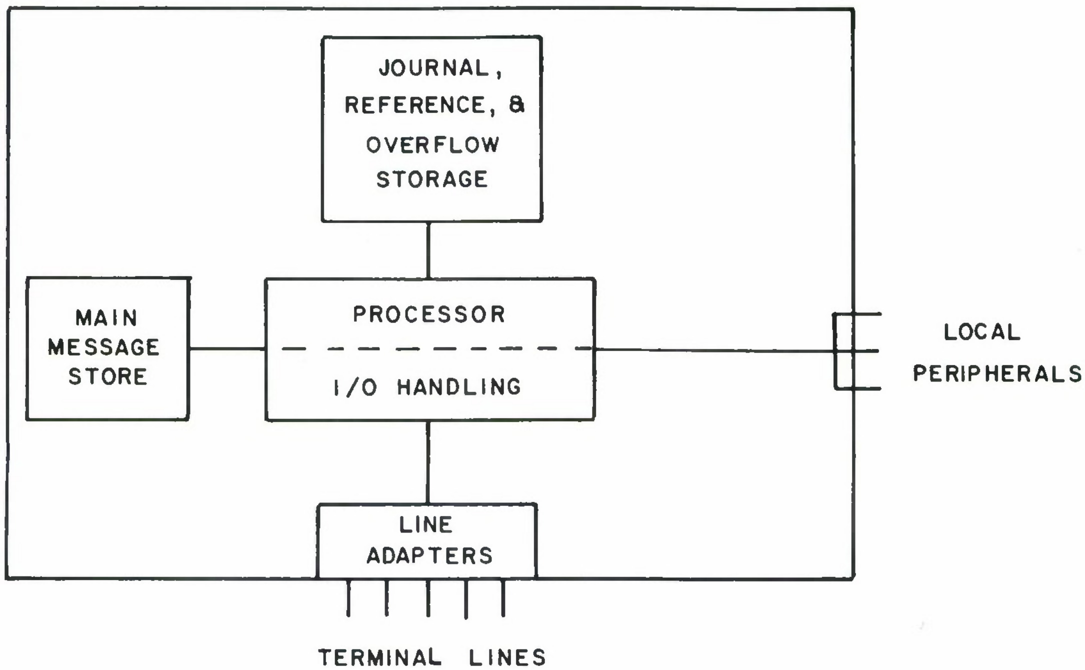
- Processor and Main Message Store - The processor handles traffic routing, provides error correction and code conversion, and controls the I/O operations among the local storage areas, as well as with other message switches. The main message store contains messages being readied for forwarding.

- Auxiliary Storage Devices - These devices, including magnetic tape or disk storage, provide overflow storage as well as journal storage for permanent records, and reference storage for temporary backup or traffic logging purposes.

- Local Peripheral Devices - These devices, such as teletype consoles, allow operator interaction with the message switch, and they provide a means for outputting status and system operation information.

- Communications Line Adapters - The adapters interface the different data lines to the message switch.

Suitable software must be provided to handle the line types, peripheral devices, message handling functions, and traffic logging requirements of the particular applications.



IA - 46,627

Figure 3 MESSAGE SWITCH COMPONENTS

Packet-Switching

A variation of the message-switching system is packet-switching. The procedures are essentially the same as those of message-switching, except each message is divided into packets of a fixed length before transmission to their destination. This allows a more dynamic assignment of output channels since each packet may be sent along the route with the shortest transmission delay at that particular moment. Each packet can then be error-checked at each switching station, and finally reassembled at the destination. Long messages are received faster with this method since each packet may be sent over a different route. This also results in a more effective use of the high-speed transmission lines, with an increased cost savings. A further explanation of packet-switching is included as part of the ARPANET description in Section III.

REMOTE CONCENTRATION

The basic purpose of a remote concentrator is to accept data from many low-speed terminals and transmit this data via a high speed line (rather than many low-speed lines), thus decreasing telecommunications costs. In addition, the remote concentrator receives data from the network and distributes it to the appropriate terminals. In its terminal handling function, the remote concentrator operates much like a front-end processor that is not local to the host; however, since fewer communications control functions are handled, the design requirements for a remote concentrator generally are less demanding than those for a front-end processor or message-switching device.

Concentration Techniques

Devices to reduce telecommunications costs can generally be considered in two different classes: the hard-wired multiplexors and the programmable concentrators. When speaking of the hard-wired devices, the distinction between multiplexing and concentrating becomes important although the two terms are often used interchangeably. Multiplexing refers to the mixing of multiple incoming data paths onto a single output channel, while concentration is the accommodation of more terminals than there are multiplexed channel slots.(3) In multiplexing, given frequency or time slots on a channel are assigned on a predetermined basis, whereas in concentration, a large number of input channels share a smaller number of output channels on a demand basis. The hard-wired multiplexors handle incoming data bit-by-bit, provide no buffering, and appear completely transparent to the user since no processing or buffering delays are incurred. The programmable concentrators handle the data at the character level before forwarding.

Characters are assembled into message blocks by the concentrator, and the blocks are then transmitted over high-speed lines. These character-level buffered concentrators are not transparent to the system, since delays are introduced by the block assembly. If devices are used which require the fast echoing of a transmitted character, then the echoing must be done at the first level of buffered concentration.

Functions

The following are the major functions of a remote concentrator:

- Line Control - The remote concentrator must sample each incoming line for an arriving bit stream. Code conversion and error checking are performed on the incoming data. The complexity of the line control function depends on the variety of the attached data line speeds and character codes.

- Message Assembly - The remote concentrator assembles a message block from the data incoming from the connected terminals. Small data blocks can be assembled into larger message blocks before transmission, increasing the efficiency of line utilization.

- Message Dissemination - The remote concentrator receives input from the network, determines the proper destination, and forwards the data.

Remote concentrator connection to a host processor can take several forms. A direct high-speed line to the host may be available, but, in general, the host FEP would interface an incoming line from the remote concentrator, or the FEP might interface several remote concentrator lines. Where transmission lines are limited, a multidrop scheme may be used, where each remote concentrator is periodically polled before forwarding its traffic. In any event, it is desirable to implement some of the line control function in the remote concentrator to alleviate the overhead of the host FEP. Error checking in the remote concentrator can isolate bad data without involving the host; this introduces another level of error checking above the host and its FEP.

Advantages of Concentration

There are three major advantages associated with the use of programmable remote concentrators:

- Reduced Line Costs - A remote concentrator can reduce the cost of communication lines. The actual reduction is a function of the number of attached lines, distance to host processor, and terminal utilization.

- Flexibility - With a programmable concentrator, the addition of remote stations would not necessitate major changes in host processor software. Since the terminal configuration software is resident in the concentrator, it can be modified to handle the specific characteristics of any new remote terminals, with minimum effect on host software.

- Control Function - A programmable concentrator can handle certain control functions, as previously mentioned.

When used strictly for line concentration, a hard-wired multiplexor utilizing time-division multiplexing is generally as effective as a programmable concentrator in reducing line costs; however, the application of programmable processors to remote concentration is a result of their flexibility. Software changes can be made to implement new configurations, and functions previously left to other processors can be handled in the concentrator.

A tradeoff must be made regarding host response time to a terminal connected via a concentrator. The hard-wired multiplexors provide transparency and virtually no delays, but at the cost of flexibility. The programmable concentrators introduce buffering delays, but provide greater flexibility. The buffered block size also has its tradeoffs. A large block favors a more efficient use of high-speed lines, while a short block favors a faster response time.

Summary

In summary, it is important to reiterate the point that although front-end processing and remote concentration are similar in many respects, they differ in their primary purposes. Front-end processing is concerned mainly with relieving the host computer of I/O handling, while remote concentration reduces communication line cost by replacing many low-speed lines with fewer high-speed lines and making more efficient use of the higher bandwidth.

SECTION III

EXAMPLES OF EXISTING SYSTEMS

This section describes existing systems which use communications processors. The aim is not to judge the design, but to present examples showing how the communications processor requirements within a system are met. For front-end processing and remote concentration, a WWMCCS configuration example is presented. For message-switching/packet-switching, an ARPANET example is used.

WWMCCS COMPOSITE CONFIGURATION

The equipment available through the WWMCCS contract has gained widespread use in military command and control system applications. A composite system using WWMCCS equipment exhibits the front-end processing and remote concentration concepts. This composite system will consist of the following devices: a Honeywell 6000 Series computer as host processor; a DATANET 355 Front-End Network Processor as the host front-end; and a remote terminal consisting of a System 700 Remote Message Concentrator/Remote Batch Configuration. The composite system is shown in Figure 4. The communications processors within the system are described below.

Front-End Processor

The Honeywell DATANET-355 Front-End Network Processor is the front-end for the H6000-series host computer. It is a communications-oriented, stored-program computer system which can be programmed to reduce the line control overhead of the host, and allow the host to process data more economically. As an FEP, the DATANET 355 receives data from remote terminals, stores it temporarily, and processes and forwards it to the host. When host processing is completed, the output is returned to the DATANET 355 for forwarding to the proper terminal. A direct access mode of operation is also possible, where the FEP simply transfers information between a program executing in the host and the terminal communicating with the program.

Documents providing detailed descriptions of the DATANET 355 are listed in the bibliography. To understand the execution of the FEP function within the DATANET 355, details of the major components are included here.

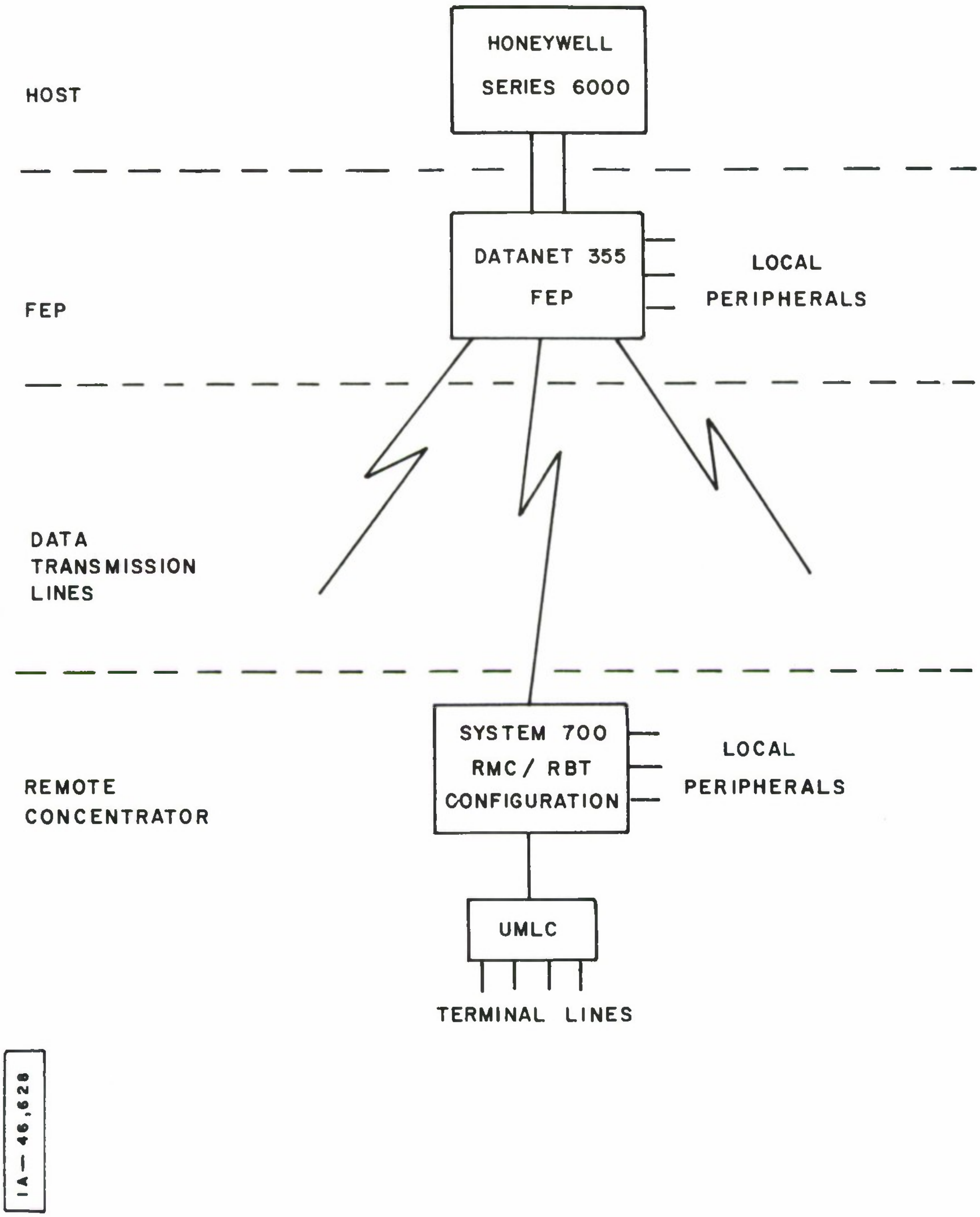


Figure 4 WWMCCS COMPOSITE CONFIGURATION

Components

The DATANET 355 consists of the major FEP components as presented in Section II. These components and their descriptions are as follows:(4)

- Central Processor - The processor is a stored-program computer with a capacity of 16K or 32K, 18-bit words. It is a parallel binary, fixed point computer using a random access memory with a 1 microsecond access time. Control instructions and control hardware are similar to the H6000.

- Intercomputer Adapter - The Intercomputer Adapter (ICA) links the DATANET 355 I/O multiplexor to the H6000. For FEP operation, the ICA transfers the data and control information between the FEP core storage and the host core storage. The ICA provides the flexibility of connecting up to four hosts to a single DATANET 355.

- Communication Line Adapters - There are two types of line adapters available:

    - The High Speed Line Adapter handles up to 32 lines. Three adapters may be attached to the FEP. Synchronous or asynchronous character-oriented terminals at speeds of 75 bits-per-second (bps) to 50 kbps, with a variety of character sizes, character sets, and information codes can be connected.

    - The Low Speed Line Adapter provides time-division multiplexing by character. Up to six adapters can connect to the FEP. The adapter supports 52 terminals at 110 bps, 26 terminals at 150 bps, and 17 terminals at 300 bps.

- Peripheral Adapters - Adapters are available for interfacing peripherals such as printers, readers, punches, tape cassettes, and disk storage systems.

Software

The FEP software in the DATANET 355 operates in conjunction with the H6000 operating system. The DATANET 355 is controlled by either the General Remote Terminal Supervisor (GRTS), or the Network Processing Supervisor (NPS) software. Both control the FEP functions within the DATANET 355, and NPS provides added capabilities beyond front-end processing.

GRTS. GRTS is an extension of the H6000 software. Under the host operating system, GRTS handles remote terminal communications with the host computer, through batch jobs entered at the remote terminals or by direct access between the terminal and a program in the host. GRTS consists of software resident in the DATANET 355 and the H6000 computer.

The DATANET 355 GRTS software, referred to as GRTS/355 Control Program, supports the following activities:

- Information from the remote terminals can be preprocessed, (for example, checked for valid passwords), input to the host, and the output can be transmitted to the appropriate remote terminals.

- In the remote batch function, the output from a job submitted through the FEP can be directed to an alternate terminal, or it can be printed at the host site.

- A mix of wideband, 2000 bps, 2400 bps, and low speed lines is possible.

- A remote terminal may disconnect from the system, and reestablish contact later to obtain any output.

- A delta configuration, with a host-FEP shared mass storage device, can be implemented.

- Direct access between a remote terminal and a host program is supported.

The H6000 GRTS software, the GRTS/6000 program, provides the software interface between the H6000 and the DATANET 355. GRTS/6000 handles input, execution, and output of jobs submitted to the host by the remote terminals, to provide the users with the same functions available at the host site.

Support software, including a macro assembly program, a simulator, and an on-line test system, is provided with the DATANET 355.

NPS. The Network Processing Supervisor is an expanded version of the GRTS software designed to provide internetting of the DATANET 355 and message-switching capability. The enhancements included in NPS are:

- Addition of message switching to the network control functions;

- Dual, redundant DATANET 355 configurations may be used with NPS;

- Supervisory control and statistical reporting functions are added to increase control of the network;

- Network configurator, to ease the adaptation of NPS to a particular system;

- Message format of each line to the DATANET 355 may be defined as necessary;

- Remote concentrator systems are supported.

Summary

The DATANET 355 FNP is used as a front-end processor for the Series 6000 Honeywell computers. It is not a candidate as an FEP for other host computer systems. Up to three high-speed line adapters, serving a maximum of 96 lines, or up to six low speed line adapters, serving up to 312 lines, can be configured on the DATANET 355. The maximum configuration is dependent on the speed of the data lines involved. The introduction of the NPS software increases the utility of the DATANET 355, and, with remote concentration, message-switching, and full-duplex operation included, the Series 6000 System has a broader range of communications applications.

Remote Concentration

The Honeywell System 700 Remote Message Concentrator/Remote Batch Terminal (RMC/RBT) satisfies the following requirements:

- Concentration is provided for several terminals, either synchronous, asynchronous, or mixed.

- The concentrated terminals appear as if they are connected directly to the host computer, although handled through the remote concentrator.

- A certain amount of pre-processing is done by the concentrator, including reformatting and data checking, prior to transmission to the host.

- The remote concentrator receives messages from the host and transmits them to the proper terminals.

- The remote batch capability using batch-type peripherals (card readers, card punches, and line printers) is available, and it operates concurrently with remote concentration.

Existing documents detailing various aspects of the System 700 are listed in the bibliography. Descriptions of the System 700 components are not included here; however, within the scope of this report, the multiline handling capabilities are detailed, as a representation of the remote concentration function.

There are three multiline controllers available with the System 700. Each is designed for differing line capacity and speed requirements.

Universal Multiline Controller (UMLC)

The UMLC interfaces both synchronous and asynchronous lines to the System 700 computer, the Honeywell 716. Since it is a character-oriented device, it assembles bit-serial data into characters for input to the processor and disassembles parallel data into bit-serial data upon output from the processor. The maximum data transfer rate is 57,600 characters per second. The UMLC can control up to 64 lines in a modular expansion of two-line groups. The maximum speeds supported are 10,800 baud for synchronous lines and 2400 baud for asynchronous lines, although these exceed the capabilities of generally available modems. As more data lines are added, the maximum speed decreases as shown in the following table:(5)

Number of Lines	Maximum Speed (Baud)
up to 16	10,800
up to 32	5,400
up to 64	2,700

This table assumes that all lines are in the same category, that is, high-speed, medium-speed, or low-speed lines. A high-speed line is between 5.4 kbaud and 10.8 kbaud, a medium-speed line is between 2.7 kbaud and 5.4 kbaud, and a low-speed line is between 0 and 2.7 kbaud. These categories define the number of times a line is tested during the UMLC scanning operation. High-speed synchronous lines are checked four times, medium-speed synchronous lines are checked twice, and low-speed synchronous lines and all asynchronous lines are checked once during a line scan. From this scanning information, one can determine the maximum speed allowable for any configuration. Some examples of this follow.

Example 1: 16 lines are to be connected to the UMLC. What type of lines may they be?

Result: The UMLC line scanning operation consists of 64 counts. Since only 16 lines are to be configured, each may be checked four times per scan, thus allowing for 16 high-speed lines. Of course, lines of lower speeds may also be used.

Example 2: A configuration currently has eight high-speed lines and eight medium-speed lines. How many low-speed lines may be added?

Result: The eight high-speed lines use 32 counts per scan. The eight medium-speed lines use 16 counts per scan. Therefore, 16 counts remain for low-speed lines, or there exists the capability for 16 additional low-speed lines.

The parameters of the attached UMLC lines are software configurable. These functions include the type of line, type of parity, character length, and data speed. For asynchronous lines, the program-configurable speed option may be one of six fixed speeds (1800, 1200, 600, 300, 150, or 75 baud), or one of two additional speeds conditioned through the use of the Programmed I/O (PIO) instructions. The UMLC includes an internal random access memory (RAM), 256 locations of 18 bits each, to hold data characters, line parameters, and control information. This controller interfaces to the Direct Memory Access (DMA) channel of the 716. When the UMLC is used for line speeds greater than 300 baud, a special Real Time Clock option is required.

Core requirements for a system operating in a remote message concentrator configuration and utilizing a UMLC for synchronous line interfacing are as follows:(6)

Active Terminals	Core Requirements
50	32K words
30	28K words
10	24K words

These values include the core necessary for the OS/700 operating system components, and core requirements would be lower for a mix of synchronous and asynchronous terminals interfacing to the UMLC. While not an exact result, the table presents general core requirements; a detailed calculation would be necessary for determining the core requirements of a specific configuration.

Each line to the UMLC requires an appropriate line module. The modules, synchronous and asynchronous, handle two lines of the same type. Line types cannot be mixed for an individual line module. A total of 32 modules may be used with the UMLC for a total of 64 lines.

Medium-Speed Line Controller (MSLC)

The MSLC is identical to the UMLC in most respects. It handles synchronous and asynchronous lines up to 10,800 baud; however, the MSLC has a 16 line capacity. Since the MSLC does not interface to the 716 through the DMA, but through the Programmed I/O bus, its peak load capability is lower than the UMLC.

Similar to the UMLC, the MSLC requires appropriate line modules for the interfaced lines. Each module handles two lines of the same type, asynchronous or synchronous. Eight line modules may be interfaced for a total of 16 lines.

Low-Speed Multiline Controller (LSMLC)

The LSMLC may be used when only asynchronous devices are to be concentrated. It is capable of a maximum transfer rate of 11,000 characters per second. As with the UMLC, it is a character-oriented device and contains a 256 x 18-bit memory. The LSMLC can control up to 128 narrow-band lines with mixed terminal speeds of 45 to 300 baud. Unlike the UMLC, functions such as baud rate, transmission code length, and the number of stop bits, are not program-configurable and must be hard-wired into the controller. The number of lines interfaced to the LSMLC can be expanded by using the line modules which interface four lines each. Thirty-two line modules may be used for a total of 128 lines.

Software

The RMC/RBT configuration uses the Honeywell Remote Network Processor (RNP) software, expanded version. This software supports remote batch and remote concentration functions, concurrently or independently. There are two versions of RNP software, commercial and WWMCCS version. The WWMCCS version supports additional peripheral devices such as the remote line printer. No disk support is provided.

Summary

The RMC/RBT configuration is an example of the remote concentration concept. This configuration is applicable when many remote terminals require communications with the host processor. The remote batch capability allows certain jobs to run in the non-interactive mode. The final determination of the components of the RMC/RBT configuration would be based on the number of remote terminals and their types.

THE ARPA NETWORK

The ARPA Network (ARPANET) interconnects many dissimilar computers in a nation-wide, resource-sharing network. While a complete system description is beyond the scope of this report, the subnet configuration which implements packet-switching within the ARPANET will be detailed.

General

The ARPANET exemplifies a message-switching network using the packet-switching concept. This network allows persons at one location to use interactively the data and programs of another location. The host computers connected by the net are not necessarily of the same type. The ARPANET developed as a communications system where a path for each message is not established in advance and each message carries an address. The net is a non-fully connected, distributed, message-switching system. This design differs from a star-connected network where a central message-switching center handles the traffic from all nodes. In the distributed network, each node has a store-and-forward center, and messages routed through the network may pass through several switching centers before arriving at their destination. At each center, a copy of the message is stored until reception by the following center is acknowledged.

Subnet

To alleviate some of the problems associated with the interconnection of many dissimilar host computers, a subnet of small processors was developed.(7) These processors are identical to one another, and they interface the host computers to the net. Figure 5 shows the subnet, within the dotted line, relative to the overall network. The subnet has a two-fold purpose: 1) the problems which arise at the host computer sites will not affect the operation of the net; 2) requirements for buffering, synchronization, error control, and the like can be handled by the subnet processors, leaving the host free for other processing. The computers in the subnet are referred to as Interface Message Processors (IMPs). The original IMPs were Honeywell 516 computers; however, as the network requirements became more defined, the less-expensive and less-powerful Honeywell H316 became the IMP computer.

Subnet Design Considerations

The subnet design for the ARPANET was developed to meet the following criteria:

- The subnet should be a communication system with a main

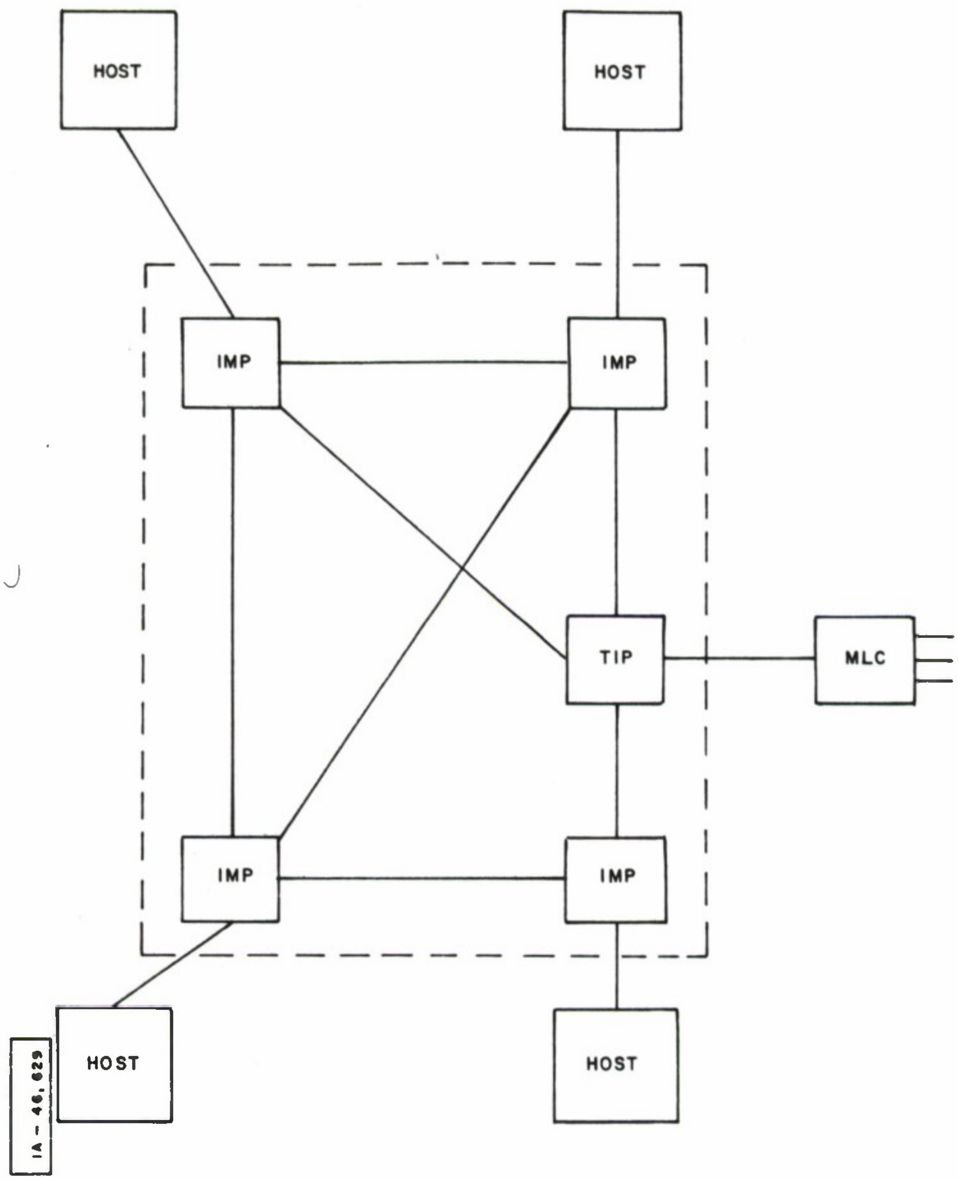


Figure 5 ARPANET SUBNET CONFIGURATION

function of transferring bits reliably from source to destination. Message reliability and error control should be handled within the subnet.

- The average transit time from source to destination should be under a half-second.

- The subnet must operate completely independent of the hosts. The IMPs must not depend on local hosts for buffering or program reloading, and the hosts must not be able to alter the IMP.

- Host-to-host protocols should not be a part of the subnet design.

Each IMP in the subnet can be interconnected via 50 Kilobit-per-second data transmission lines to a maximum of five other IMPs. An IMP can also service up to four host computers, by means of individual IMP-host interfaces. This interface is divided into two parts:

- The standard part, which is built into the IMP and is identical for all hosts;

- The specialized part, tailored to the requirements of each host.

This interface is bit-serial, full duplex. Being bit-serial, the interface can more easily handle the various word lengths of the host computers. The specialized part of the interface operates asynchronously using a ready/transmit procedure. This allows the bit rate to adjust to that of the slower device, and the interface does not have to meet timing specifications. The three types of interfaces available are:

- Local interface - for interfacing the IMP to a host located within 30 feet.

- Distant interface - for interfacing the IMP to a host located up to 2000 feet away.

- Very distant interface - for interfacing the IMP to a host located at distances greater than 2000 feet, and requiring the use of modems.

IMP Hardware

The Honeywell H316 minicomputer is the processor used in the latest IMP configurations. The earlier IMP design used the Honeywell 516 computer, but, as the network developed, the higher bandwidth of the 516 was not required at many locations, and it was decided to use the H316 in the IMP design. The H316 can provide up to two-thirds the bandwidth at about one-half the cost of the 516.

The H316 is a general purpose minicomputer with a maximum main memory size of 32K, 16-bit words. The H316 has a cycle time of 1.6 microseconds, versus .96 microseconds for the 516. Both computers are logically identical, and they employ the same basic instruction set, allowing the 516 software to be run on the H316. The characteristics of the H316 required for the IMP are:

- 16K words of core memory;
- 16-bit word length;
- real time clock;
- 16 subchannel Data Multiplex Control (DMC);
- 16 channels of priority interrupts.

IMP Software

The IMP software was developed to satisfy the following requirements:

- segmentation of host messages into packets;
- receiving, routing, and transmitting of store-and-forward packets;
- retransmitting unacknowledged packets;
- reassembling received packets into messages for the host;
- generating acknowledgements and other control messages.

Terminal Interface Message Processor (TIP)

As the ARPANET developed, a need was recognized for a configuration whereby users who had no host computer of their own could access and utilize the resources available within the network. One possibility was to allow a user to connect into a nearby host. Through the host, the other resources of the net would then be available. This configuration would require the host to possess the necessary terminal handling capabilities. The user would also be dependent on the host for continued access to the net.

As an alternative to direct host connection, an IMP was developed capable of handling a variety of terminal types. This design is referred to as the Terminal IMP or TIP.(8) The TIP was designed as a flexible multiplexing device, not capable of being programmed from the terminals. The required storage and computational capabilities would be in the hosts or in the terminals, not in the TIP.

The TIP is a single computer containing the standard IMP programs with special routines added to allow the TIP to interface the terminals to the host computers in the net. Similar to the IMP, the TIP is built around a Honeywell H316 computer; however, additional core storage and a multiline controller are included. With the special multiline controller, the TIP can handle up to 63 local or remote terminals of various characteristics. Since no interpretation of the data is done, any character set or character length (5, 6, 7, or 8 bits) is acceptable. Data format is bit-serial, using start bits and one or more stop bits, and operation may be full or half duplex. The controller can handle devices such as teletypes, alphanumeric CRTs, modems, and peripheral devices such as card readers, and line printers, with additional remote interface units. Since the TIP contains the essentials of the IMP software, up to three host computers may connect to the network through the TIP.

Message Handling

In the ARPANET, the IMP takes messages from the host, and converts them into "packets" of 1008 bits. Each packet is then sent separately into the network towards its destination along the best available route at the given moment. By this technique, packets within the same message might not necessarily take the same path to the destination, since each path is not predetermined. It is also possible for the packets of a message to arrive at the destination IMP non-sequentially. The destination IMP must reassemble the packets into the message for delivery to the host as a single unit. The packet transmission is completely transparent to the hosts.

Any IMP will store a message packet until a positive acknowledgement of receipt is returned from the succeeding IMP. This procedure continues until the destination IMP receives the packet. An IMP may refuse a particular packet by failing to return a positive acknowledgement. This may be necessary if the packet was in error, if the IMP is busy, or if buffer storage is temporarily full. If the packet is not acknowledged within a specified time (such as 100 msec), the transmitting IMP will send the packet again, possibly over a different route. Buffer storage limits the maximum size of a message to 8095 bits. Messages may be of varying lengths and formats. The host must indicate the end of the message to the subnet.

During the early operation of the ARPANET, an interesting problem in packet-switching appeared in the net.(9) In peak periods, messages arrived at the destination IMPs faster than they could be forwarded to the hosts, causing reassembly congestion. As the IMP's backlog of traffic increased, reassembly lockup resulted when the destination IMP was unable to pass any traffic to the host.

Reassembly lockup occurs when the IMP has dedicated all its reassembly buffers to specific messages, parts of which are already in the buffers. At this point, the IMP can only accept packets destined for those messages. When the packets required for message completion are further than one node away, lockup results. The destination IMP cannot accept packets for messages other than those in the buffers, yet by not accepting and acknowledging the incoming packets, no subsequent packets would be forwarded. Thus, the messages in the reassembly buffers would never be completed or forwarded to the host.

This problem was solved by utilizing the buffer storage at the source and destination IMPs. Any multipacket message entering the network must have sufficient storage allocated at the destination IMP. After taking the first packet of a message from the host, the source IMP sends a control message to the destination IMP requesting that reassembly storage be allotted. The destination IMP will return an allocation message when storage becomes available, and the source IMP will then forward the message packets. In actual operation, the request mechanism is incorporated into the message acknowledgement sequence to allow full use of network transmission capabilities and to reduce the individual control messages an IMP must send. Reassembly lockup is prevented because the destination IMP will have the required storage allocated for each incoming message to its host.

SECTION IV

TECHNOLOGY IMPACT

With the move towards a standard Data Link Control Procedure, the expansion of microprocessor applications, and the potential development of a standard Higher Order Language for communications processing, future network design considerations may differ greatly from those of the past. The state of development of the different technologies and their potential impact on communications processors and network design are examined.

DATA LINK CONTROL PROCEDURES

A Data Link Control Procedure, or line protocol, is an established sequence of programmed events to control communications among network users in a well-defined and concise manner. Being considered here are the more sophisticated data communications techniques characteristically used with higher speed transmission facilities. There are several such protocols currently in use; each application of a particular protocol is often a function of the hardware on which it operates. Some manufacturers have developed their own protocols to overcome the inefficiencies of the older versions; attempts to standardize the different procedures and produce a universal protocol are also being undertaken.

General Functions

The functions of a line protocol include the following:

- Data Transfer Control - This is the foremost function. A protocol allocates certain fields within a transmission block for specific control information. These fields are identified by control characters, such as for a start and end of text; however, the latest protocols indicate only the start of a block and utilize control fields of fixed length within the block.

- Error Checking - Several techniques are used to assure correct data reception:

    - Vertical Redundancy Check (VRC) checks each character as received, using a simple parity scheme.

    - Longitudinal Redundancy Check (LRC) checks an entire block by generating a Block Check Character resulting from an "Exclusive OR" operation on all bits in the block.

- Cyclic Redundancy Check (CRC) checks the entire block using a division of a polynomial (the data bits as coefficients) with a constant polynomial.

- Synchronization - Some protocols precede the data block with special characters signalling the start of message.

- Transparency - This is the mode in which binary data is being transmitted and any control character bit patterns appearing in the data stream must not be acted on. This is achieved in one of three ways:

  - Character Stuffing - when a control character pattern is sent as data, an additional character is added to signal the receiver that the bit pattern is data.

  - Bit Stuffing - similar to character stuffing, but concerned with flag bit pattern. Bits are inserted when a flag pattern appears as data.

  - Count - a count of the data bytes in the block is provided.

- Line Utilization - Depending on the protocol, half-duplex, full-duplex, or either transmission mode may be possible.

Highlights of the differences among the most popular protocols, and how they handle the basic functions follow.

Binary Synchronous Communication

One of the most popular link protocols in use today is the IBM Binary Synchronous Communication (BSC). This protocol was developed for IBM computers and terminal equipment, but it has become a de facto industry standard supported by many manufacturers' devices. It is a half-duplex protocol, using control characters to designate the various fields within a block. The error checking technique used depends on the information code being sent: for ASCII, VRC is used on individual characters, with LRC on the entire block; for EBCDIC or 6-bit Transcode, the CRC is used. In the transparency mode, a character-stuffing technique is used to avoid the unintentional decoding of a control character.

The greatest drawbacks of the BSC protocol are its half-duplex operation and its requirement for an acknowledgement sequence following each transmitted block. These drawbacks and the desire to increase in communications resource efficiency have led to the development of the more efficient full-duplex protocols.

Synchronous Data Link Control

IBM developed the Synchronous Data Link Control (SDLC) to meet the expanded requirements of a link protocol in a simple and straightforward manner.(10) SDLC is intended for bit-serial synchronous communications between buffered stations on a data transmission link using centralized control in either full or half duplex mode. The major features of SDLC are as follows:

- bit-oriented and independent of character code structure;

- only control character is "flag" which frames the message;

- CRC error check performed on entire message, not just information bits;

- bit stuffing performed to prevent unintentional flag characters;

- employs a sophisticated frame sequencing scheme which provides the transmitted frame number as well as the number of the last successfully-received frame;

- allows up to seven outstanding unacknowledged frames.

Being bit-oriented and independent of code structure, the protocol can achieve a higher degree of flexibility to help improve throughput.

Digital Data Communications Message Protocol

Digital Equipment Corporation (DEC) has developed its own protocol designed to operate on existing hardware, the Digital Data Communications Message Protocol (DDCMP).(11) DDCMP is a character-oriented protocol operating on synchronous or asynchronous, full or half-duplex links. The major features of DDCMP are as follows:

- ASCII control characters used to specify type of message;

- compatible with existing DEC hardware;

- uses two CRC error checks, one on header and one on data;

- a count field specifies the length of the data field up to 16K bytes;

- maximum of 255 messages may be received before acknowledgement is required.

The ability of DDCMP to operate on synchronous, asynchronous, or parallel transmission lines, and its compatibility with existing hardware are two of the most important features. One drawback is the inefficiency when handling short messages, since a 10-character control overhead is required for every message.

Standardization

Unlike BSC, the latest link protocols are capable of both full and half-duplex operation. Standards organizations, in particular, the American National Standards Institute (ANSI), have been working on a standard protocol which would contain many of the features of the manufacturer-generated protocols. The proposed ANSI versions, the Advanced Data Communications Control Procedures (ADCCP), parallel SDLC more closely than DDCMP. While all are full or half duplex, DDCMP can operate in an asynchronous mode, while SDLC and ADCCP cannot. Another important difference is the method of achieving data transparency. Since DDCMP is a character-oriented protocol, it provides a count of the data field length, whereas the others, being bit-oriented, utilize the bit stuffing technique. Due to this difference, SDLC and ADCCP require bit stuffing hardware, while DDCMP can operate on many existing configurations.

Impact

Since standardization of link protocols is still in the proposal stages, it may be some time before the implementation of a single standard link protocol occurs on a widespread basis. Although major standard efforts are towards a bit-oriented protocol, character-oriented schemes will remain operational for the following reasons:

- A character-oriented protocol is suitable for asynchronous line operations.

- The effect of the higher control character overhead of these protocols is reduced in systems in which long messages are common.

- Many features of a new protocol may be hardware-implemented, possibly providing switching from bit to character-oriented protocols as needed. Devices are currently being developed to allow switching from BSC to SDLC.(12)

Standard protocols would ease the requirement that communications processor software convert from one protocol to another. This would reduce the complexity of the software, with a corresponding decrease in development time and cost.

The standards development could result in different protocols, each a "standard" for a particular situation. Bit-oriented protocol standardization is ongoing, and some development has been done on a character-oriented protocol standard by the International Organization for Standardization.(13) The impact these standards have on existing systems will be lessened as hardware devices become available to handle conversion from existing protocols to the standard, or from one protocol orientation to another.

MICROPROCESSORS AND MICROCOMPUTERS

A microprocessor generally consists of one or more large scale integration (LSI) chips containing the arithmetic logic unit, I/O control logic, and general-purpose registers. A microcomputer is formed when memory and I/O devices operate with a microprocessor. The interfacing of a microprocessor can require considerable effort; consequently, the prepackaged microcomputers, with their interfaces for peripherals, memory and control devices, have become increasingly popular. Microcomputers are potential replacements for minicomputers in a variety of applications. How microprocessors and microcomputers compare to the minicomputers, and the potential application areas are discussed.

Advantages and Disadvantages

The state-of-the-art in microprocessors has been changing dramatically in recent years, and microprocessor applications are continually expanding. When considering microcomputer replacement of a minicomputer, certain advantages and disadvantages of current designs are apparent.(14) The advantages include:

- Smaller Size - Due to LSI technology, a programmable microcomputer can be contained in a package as small as two inches square.

- Reduced Power Consumption - An LSI microcomputer uses from 66 to 75 percent less power than a comparable minicomputer.

- Lower Cost - When considering CPU costs, a microprocessor results in substantial cost savings; however, the memory and peripherals used to form a microcomputer do not yield any substantial cost savings over the minicomputer equivalents.

- Higher Reliability - The microprocessor reliability and the reduced number of boards, necessitating fewer interconnections, increase system reliability.

The disadvantages include:

- Slower Speed - A microcomputer may be as much as an order of magnitude slower than a minicomputer.

- Word Length - The currently available maximum word length is 16 bits, although bit-slice technology has provided greater lengths for specialized applications.

- Memory Size - Maximum of 64K words of memory.

- Interrupt capability may require add-on chips.

The software provided varies among the available devices, and certain packages may not be available on some designs. For most applications, an assembler or cross-assembler is necessary to ease the programming burden. A debug package or simulation software should be available for testing the programs. If a microcomputer is used for large applications, where it may be required to handle the loading and unloading of different programs from peripheral storage devices, an operating system is desirable. Some microcomputers are available with a compact version of PL/I, but most current microcomputer applications require small programs, and the coding time saved by a high-level language may be more than offset by the increased memory and running time requirements of the program.

Peripheral devices interfaced by the microcomputers include teletypes, paper tapes, card readers, and analog interface systems. Some microcomputers interface mass storage devices, but to keep the device cost within the microcomputer range, these have been limited to floppy disks and cassette storage systems.

Impact

The initial impact of microprocessors and microcomputers will be to increase the distributed processing capability of a network. Super-intelligent terminals, peripherals with a built-in processor, and intelligent modems will allow more localized data processing, with the network providing access to centralized data bases. The demands on centralized host computers for processing support will be reduced.

Microprocessors could off-load some of the control functions presently handled by the communications processors. Separate modules could be used for functions such as code conversion and error checking. For example, functions traditionally handled by a remote concentrator might be microprocessor-implemented at each terminal, permitting a direct interface to a host FEP; however, the increased polling load of the FEP in such a case would have to be considered.

A secure terminal, one where the cryptographic hardware is built into the terminal, may be developed as a result of microprocessor technology. For secure communications using existing hardware, data from each line must pass through a crypto device prior to transmission to the host. In addition, present cryptographic devices are costly. Using microprocessors, the cryptographic hardware could be built into any terminal. At the host site, a multiplexor containing matching crypto hardware could accept data from many secure terminals, decrypt it, and forward it to the host FEP. The advantages of a microprocessor design would be in the reduced cost and the increased flexibility. In theory, any terminal would be capable of secure transmission by adding the microprocessor-based cryptographic device. Under a current advanced development program, Fairchild Corporation will explore the use of microprocessors in the secure terminal and remote multiplexor, and produce a limited number of prototypes.

The advent of microprocessor technology has made the design of a secure FEP more feasible. Modularity would be the key to this implementation. Each software function would be identified, proven functionally correct, and executed by a microprocessor module. The development of the proven-correct software will be simplified by the modular design and implementation. Any subsequent changes to the FEP could be more easily accomplished by changing only specific hardware/software modules, without altering the secure characteristics of the other fully isolated modules.

The direct replacement of minicomputers with microcomputers has not occurred on a large scale. The slower speeds of the microcomputers, along with the long development time required for a microcomputer-based system, have moved emphasis towards the distributed processing applications. Minicomputer replacement should become more significant as the microcomputer capabilities continue to approach those of present minicomputers. These conclusions parallel those of a recent report(15) on the future of data communications.

HIGHER ORDER LANGUAGE STANDARDIZATION

The expansion of computer networks in recent years has led to increased interest in the development of a standard higher order language (HOL). There are two partially-overlapping aspects in the development:

- The development of a standard programming HOL concerned mainly with operation within a single computer, i.e., not handling network procedures to any degree.

- The development of a networking language to standardize not only local computing operations, but also the passing of data through the network.

Standardization efforts have been undertaken to develop a programming HOL for use in Air Force communications systems. The JOVIAL language is the programming HOL most widely used by the Air Force although no standard exists. The major advantages of a standard programming HOL would be transferability from computer to computer, better compiler development, ease in documentation, and standardized support.

Development has also begun on a high-level networking language, which would handle tasks such as data coding, I/O control, and data manipulation. Since no existing language is suitable as a standard, a new language must be developed. Although beyond the capabilities of the existing languages, the new language should ideally incorporate a number of network functions: (16)

- determine what network resources are needed;

- establish the network connection;

- handle system protocols;

- supervise execution of each task;

- terminate connection when operations are completed.

A single standard language may not be suitable for every network configuration. A number of "standard" languages may be developed to meet the different requirements. To ease implementation, the language should have a similar structure to existing high level languages, the network features must be totally transparent, and the language must also operate in a non-network configuration. The problems of incorporating numerous requirements into a single language may result in the initial development of standard languages which operate on only very limited and specific configurations.

Impact

The acceptance of a standard programming HOL for use on all computers within a network would provide software transferability within the network. When considering the addition of a new computer to the network, concern over its capability to execute existing software will be lessened considerably. As a result, more heterogeneous networks would result, with each host computer capable of using the software of any other. A standard networking language would further ease the problems of connecting heterogeneous computers in a network, to the point where a computer could more easily connect to many different networks, using the resources of each as the need arises.

Since there are many higher order languages currently in use, the change to a standard is a long and momentous undertaking. Consequently, only preliminary standardization work has occurred, and it will be some time before any large impact is felt on existing network configurations.

SECTION V

SPECIFIC DEVICE CONSIDERATIONS

There are numerous considerations involved when evaluating various communications processors for a particular application. This section will present some general rules which illustrate how the field of choice may be lessened, followed by specific device examples to identify some of the tradeoffs necessary in the selection process.

GENERAL SELECTION CRITERIA

The first step in the selection process is to identify the type of problem to be solved. A communications application can generally be placed within one of the three major areas: front-end processing, message-switching, and remote concentration. Multiline controlling is sometimes considered a fourth area, but, for this discussion, it is considered within the concentration heading. By identifying the problem as being within one area, the specific devices suitable for use can be reduced. There will often be a number of possible solutions available. Although communications processor surveys might create the impression that any of a number of devices can be selected off-the-shelf for an application, the host system capabilities and the teleprocessing features supported often dictate the suitable device. Futher examination of selection tradeoffs is necessary; these tradeoffs include:

- Cost versus performance - high line speeds and high processor speeds would bring an increase in performance, but at a similar cost increase. The expected line utilization and throughput requirements must be carefully considered, since the higher speed services would be cost-effective only if fully utilized.

- Peripherals desired - the availability of on-line peripheral devices may differ from one communications processor to another.

- Number and types of lines required - this is an important consideration when choosing the basic device model. Generally, different models of a specific device will be available to meet a variety of line requirements.

- Software desired as well as needed - considerable variation exists in the capabilities of software provided with a device, and in the alterations required to existing host software for front-end processor applications. Items such as operating systems, assemblers, terminal handlers, and host software alterations may not always be available, or, if available, they may incur additional cost. Careful consideration must be given to the software needs, and how to satisfy those needs.

Once the general requirements have been identified, specific devices may be examined for compatibility.

DEVICE EXAMPLES

Currently-available communications processors illustrate where tradeoffs can be made and how requirements are met. These examples are intended to show available capabilities, and they do not necessarily present the best choice within an application area. The examples are divided into two categories of communications processors based on hardware architecture: Limited Application, and Minicomputer-Based.

Limited Application Communications Processors

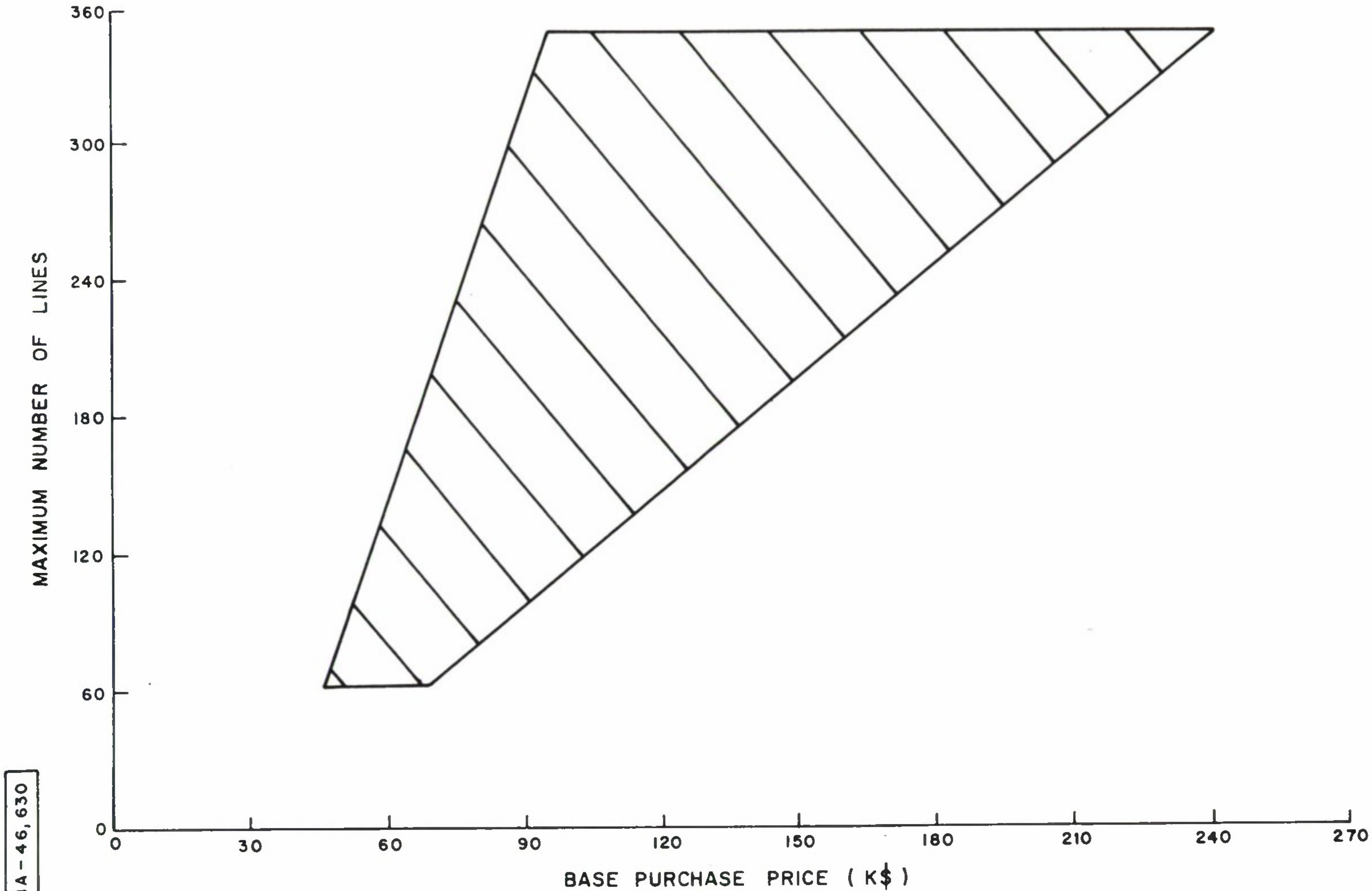
Limited application communications processors are those whose central processor is an integral component of the device, generally not identified as a separate entity, and the device is designed for a specific application. The IBM 3705 is a popular front-end processor in this category. The criteria needed to select a specific 3705 configuration are identified.

The IBM 3705 is a programmable front-end processor for a host IBM 360 or 370 computer system. It is not a candidate for front-end processing with other host computers. It can operate in two modes:

- Emulation of the functions of the IBM hard-wired controllers (270X series).

- Preprocessing and off-loading communications control functions when configured to run Network Control Program software in a System/370 virtual storage system.

The choices available when selecting a specific IBM 3705 are extensive. Twenty basic models are available, ranging from 16K bytes to 240K bytes of core storage. From 2 to 352 lines can be attached, depending on line type and 3705 configuration. Figure 6 illustrates how the base purchase price of the 3705 is a function of the maximum number of lines supported. For any of the line capacities, a range of cost exists. Although an important factor, the number of lines does not by itself indicate an appropriate 3705 configuration. To choose a single configuration, the line characteristics must be identified. These include the type of protocol (SDLC or BSC), line speed, system arrangement (point-to-



IA-46,630

Figure 6 IBM 3705 LINES VS. COST GRAPH

point, multidrop), and transmission mode (full or half-duplex). With this information, the type of Line Set can be chosen. To complete the configuration of a 3705, devices such as Line Interface Bases, Channel Adapters, and Communication Scanners must also be specified. The following steps summarize a simplified procedure for selecting the overall configuration:

- The number and types of lines are identified. From this, the necessary Line Sets can be found. Line Sets may either handle one or two lines each, and they provide the hardware interface between the communication lines and the Line Interface Bases.

- The selected Line Sets will dictate the type and quantity of Line Interface Bases (LIB) required. The LIBs provide control functions for particular types of lines and each can handle as many as eight Line Sets (16 lines).

- Channel adapters and communication scanners further affect the performance and configuration availability. The channel adapter is chosen based on required transfer rate and the method of interfacing to the host. A low-speed channel adapter operates in the network control or emulation mode. It has a transfer rate of 16 kbytes-per-second, and it attaches to the byte multiplexor channel of the 360 or 370. The high-speed channel adapter interfaces to a 370 byte multiplexor, block multiplexor, and selector channels, and it has a transfer rate of 276 kbytes-per-second. It operates in the network control mode only. A high-speed adapter with the optional two-processor switch supports 370 multiple processor configurations. The scanner chosen is based on the number and types of data lines.

- Having identified the required attachments, a minimum satisfactory 3705 model can be chosen. A larger model may be selected to provide additional memory, perhaps for future expansion.

This procedure shows the numerous options which appear when selecting a front-end processor configuration. Of primary concern is selecting a model which can support the required lines. Depending on the attachments selected, the final cost of a 3705 may be more than double that of the base model.

The software available for the 3705 includes:

- Network Control Program - for front-end operation where the 3705 provides preprocessing and control of the data communication lines.

- Emulation Program - for operating the 3705 as a replacement for the 270X series hard-wired controllers.

The software is generated within the host computer and loaded into the 3705 via its interface to the 370 channel.

An alternative to the 3705 is the IBM 3704. The 3704 can accommodate the same line types as the 3705, but a 32 line maximum exists. Similar software support is provided for the 3704, but its limited memory size (64K bytes maximum) limits the number of lines and programming features that can be accommodated at any one time. The IBM 3704 is a lower cost option when only a few lines are needed.

Minicomputer-Based Communications Processors

Minicomputer-based communications processors are built around a general purpose minicomputer, that is, a minicomputer used in various applications other than communications processing, such as stand-alone processing. While devices such as the IBM 3705 are geared to one specific application, a minicomputer-based device can generally be adapted to a variety of situations. As an example, the Data General series of communications processors is cited. This series revolves around the NOVA and ECLIPSE model minicomputers, with applications in front-end processing for IBM 360/370, network processing, message-switching, and line concentration. Application flexibility is the major feature of this typical system; the software supplied is adapted to handle the specific requirements.

The advantages of minicomputer-based configuration over a limited application communications processor include the following:

- The minicomputer configuration most often has a price advantage. Consider a Modular Computer MODCOMP System versus an IBM 3704. At comparable cost, the MODCOMP system provides greater memory (128K bytes versus 64K bytes), greater line capacity (256 to 32), greater speed (up to 3 times greater), and more types of terminals attached (CDC, Univac, in addition to IBM).

- The minicomputer configuration can often adapt to several applications. The Data General series is an example.

- Stand-alone processing capability is available. Since a minicomputer is the heart of the configuration, it could be used to handle tasks such as batch processing. This use is generally not cost effective, but it provides a back-up capability.

- Peripheral devices are more easily attached.

The major disadvantages are the following:

- Since the communications processor usually differs from the host, software generation requires more effort. The software package must often be customized to a particular installation, and when host software changes occur, the necessary changes in the communications processor may be difficult to implement.

- Service may be harder to obtain for mixed systems. This is especially true when related failures occur in two different manufacturers´ devices.

These factors must be carefully considered when choosing the best device for an application. For situations where an application is precisely defined and limited, a limited application device such as the IBM 3705 may be appropriate. When flexibility of applications is desired, such as capability of emergency processing, a minicomputer-based configuration could be the answer.

SUMMARY

This section presented the considerations necessary when choosing a communications processor. A clear definition of the application is the major criterion required before undertaking the selection process. Once the application is defined, extensive information is available describing the suitable devices. Numerous surveys detailing device capabilities have been published and a selected listing is included in the bibliography. These references should be consulted for further information on a specific manufacturer´s model.

SECTION VI

CONCLUSION

This report provides the reader with the background necessary to determine where a communications processor might be used. While not naming specific devices for each application, tradeoffs and guidelines have been presented which can be used to narrow the field of choice for a particular application. Using this information, an individual should be able to evaluate proposed concepts and designs in communications processors.

REFERENCES

1. C. J. Ball, "Communications and the Minicomputer", Computer, Vol. 4, Sept./Oct. 1971.

2. L. J. Hardeman, "Communications Processors Pace Growth in Data Network Traffic", Electronics, 24 May 1973.

3. DATAPRO Research Corporation, "All About Multiplexors and Concentrators", Delran, N. J., November 1973.

4. Honeywell Information Systems, Inc., "Series 600/6000 DATANET 355 Front-End Network Processor", BS-03, Waltham, Mass., December 1973.

5. Honeywell Information Systems, Inc., "System 700 Type 6322 Universal Multiline Controller", AG-70, Waltham, Mass., April 1972.

6. Honeywell Information Systems, Inc., "DATANET 700 RNP Type RCP 707 Configuration", AP-12, Waltham, Mass., August 1974.

7. F. E. Heart, et. al., "The Interface Message Processor for the ARPA Computer Network", Proceedings of the AFIPS Spring Joint Computer Conference, 1970.

8. S. M. Ornstein, et. al., "The Terminal IMP for the ARPA Computer Network", Proceedings of the AFIPS Spring Joint Computer Conference, 1972.

9. J. M. McQuillan, et. al., "Improvements in the Design and Performance of the ARPA Network", Proceedings of the AFIPS Fall Joint Computer Conference, 1972.

10. J. R. Kersey, "Synchronous Data Line Control", Data Communications, May/June 1974.

11. E. V. Stelmach, "Introduction to Minicomputer Networks", Digital Equipment Corp., Maynard, Mass., 1974.

12. "Newsfront", Data Communications, March/April 1975.

13. S. M. Harris, "Data Communication Standards at the International Conference on Computer Communication ´74: A Post-Conference Report", The MITRE Corp., M74-251, Bedford, Mass., December 1974.

REFERENCES (CON'D)

14. "Technology Profile on Microprocessors and Microcomputers", Modern Data, January/February 1975.

15. H. R. Karp and G. Lapiders, "What the Future has in Store for Data Communications", Data Communications, May/June 1975.

16. H. Z. Krilloff, "A High Level Language for Use with Multi-Computer Networks", Proceedings of the AFIPS National Computer Conference, 1973.

BIBLIOGRAPHY

WWMCCS Configuration:

Auerbach Publishers, Inc., "Honeywell System 700", Report No. 180.3980.400, Philadelphia, Pa., December 1972.

Surveys:

Auerbach Publishers, Inc., "Communications Processing Equipment", Report No. 715.0000.200, Philadelphia, Pa., April 1975.

Datapro Research Corp., "All About Communications Processors", Report No. 70G-400-01, Delran, N. J., October 1974.

Hobbs, L. C., and McLaughlin, R. A., "Minicomputer Survey", Datamation, July 1974.

Murphy, J. A., "Available Mini Mainframes", Modern Data, August 1974.

Staff Report, "Communication Processors: Who, What, and Why?", Telecommunications, May 1973.

Teener, M., and Liles, W., "Microcomputers", Modern Data, February 1975.

Theis, D. J., "Microprocessor and Microcomputer Survey", Datamation, December 1974.